MIL-STD-498 versus DOD-STD-2167A ISSUE PAPER

BACKGROUND

In 1991, the Joint Logistics Commanders (JLC) established the Harmonization Working Group (HWG), consisting of 50 individuals from the software community, to develop a replacement for the Defense System Software Development and Documentation Standard, DOD-STD-2167A (2167A) [1]. The HWG was tasked with four basic objectives. The initial objective of the HWG was to develop a standard which would resolve the concerns and issues cited by users of 2167A. An additional objective of the group was to merge the existing DOD Software standards, such as DOD-STD-7935A (AIS Documentation Standard), DOD-STD-2168 (Defense System Software Quality Standard), and DOD-STD-1703 (National Security (NS) Product Standard), into one single development standard which would cover all of DoD. The third objective of the HWG was to ensure compatibility with current DoD directives, instructions, and other standards, and finally the HWG was tasked to develop a standard which would serve as the basis for the US implementation of applicable portions of International Standard 12207 (ISO/IEC 12207). ISO/IEC 12207 is the current commercial standard for the entire software life-cycle process, which includes the acquisition, supply, operation and maintenance processes [2].

As a result of the HWG's efforts, MIL-STD-498 (498), the current Software Development and Documentation standard, was completed on August 15, 1994. It was approved by the Office of the Secretary of Defense on November 8, 1994 for an interim period of two years [3]. At the end of the two year period, it is planned that a commercial standard based on 498, IEEE 1498/EIA 640 (Institute of Electrical and Electronics Engineers/Electronics Industries Association), will be available. The US Navy and the US Air Force have issued service-wide waivers permitting 498 to be invoked on all contracts. This issue paper will address the changes anticipated due to the use of 498.

DISCUSSION

Provided below is a list of the 20 key issues/shortfalls associated with utilizing 2167A and its Data Item Descriptions (DIDs) that 498 has attempted to address [4].

1) Removal of the perceived waterfall development process preference
2) Improved compatibility with ADA and other object oriented methods
3) Removal of the emphasis on preparing documents
4) Accommodation of Computer Aided Software Engineering (CASE) tools
5) Improved links to system engineering
6) Improved usage of management indicators
7) Pre-tailoring by software category
8) Improved coverage of modification, reuse and reengineering
9) Increased emphasis on software supportability
10) Improved evaluation and review criteria

11) Clarification of the distinction between requirements and design
12) Improved coverage of database development
13) Elimination of the SW quality assurance vs product evaluation confusion
14) Improved usage on data intensive systems
15) Clarification of the applicability to "in-house" development and subcontractors
16) Development of a mechanism which allows for software to be ordered via a CDRL
17) Clarification of software configuration management
18) Improved compatibility with incremental & evolutionary development methods
19) Elimination of inconsistencies and holes in the Data Item Descriptions (DIDs)
20) Decreased dependence upon formal reviews and audits

The general consensus among the software community is that 498 corrects the problems identified with 2167A, and reflects and/or accommodates many of the state-of-the-art advances being made in software development [5]. 498 allows for deletion and modification of non-applicable requirements set forth in 2167A. It encourages the use of computer aided software engineering (CASE) technology and no longer explicitly mentions certain activities set forth by 2167A (such as Formal Qualification Test), although similar types of activities are described. The new standard is applicable to different types of systems (AIS, MCCR ...) and encourages the reuse and reengineering of existing software, to include existing design, architecture and coding. The emphasis on formal documentation is removed, allowing for the contractor to provide information in the format gathered within the facility. This is evidenced by the reduction in Data Item Descriptions (DIDs) from the previously mandated 52 to the current quantity of 22. The conversion from "preparing documents" to "defining and recording" information is emphasized as one of the greatest possibilities for cost savings. 498 also requires a process improvement system, such as that developed by the Software Engineering Institute. Stress testing (i.e., testing the software until it fails) is replaced by a requirement to specify system and software behavior at and beyond the expected limits of the software. Basically, 498 allows the "developer" to determine what the development approach should be, while protecting the "acquirer" by specifying performance requirements. More detailed discussions of each of the twenty issues cited above and the associated implications of these changes can be found in Attachment A.

Since 498 was designed to be tailored and does not provide a "default" development process to follow, the skill level required to use it is considered to be higher than that required to utilize the old standard. A spokesman at the US Strategic Command J66 Program Office stated that the new standard "requires more mature developers and more trust between acquirer and developer" [6]. He goes on to state that "they anticipate working with developers that are stable and have a track record" [6]. The Department of the Air Force Science and Technology Support Center recommends tailoring the level of documentation and formal review to the contractors' specific strengths and weaknesses, but only for those contractors considered to be SEI Level 3 or higher [7].

IMPACTS TO DATE

The AF's Software Technology Support Center states that program specific management documentation may be reduced by up to 64% due to the utilization of 498 [7]. Based on the REVIC cost model, where documentation accounted for seven percent of the software development effort (SDR-FQT), this translates into an approximately five percent reduction to the overall development effort. However, this only addresses the "potential" savings due to data reductions, vice actual program savings. Although the October 1995 CROSSTALK does cite a few examples of ongoing programs which are utilizing 498 to upgrade or develop database documentation [6], the NCCA SW team is not aware of a completed embedded program which adhered to 498; however, there are a few which are scheduled for completion by the end of calendar year 1996. NCCA's literature search failed to uncover any other quantitative assessments, based on either engineering judgment or historical data, of the impacts of 498.

Provided below is a synopsis of one project's first time and ongoing "qualitative" experience with the application of 498 [8]. The program consisted of a) two Configuration Items (CIs), b) reused and COTS software, c) two non-waterfall life cycle development models for the CIs, and d) a desire to minimize time wasted in traditional formal reviews and formal documentation, yet assure a maintainable system. The program office identified the four areas, detailed below, which it felt would be most impacted by the implementation of 498: 1) Integrated Product Teams, 2) Reviews, 3) Documentation, and 4) Development Approach.

Integrated Product Team

The acquirer has encouraged a positive IPT atmosphere. An aggressive independent verification and validation (IV&V) agent was assigned to help review the deliverables and periodically visit the developers in "working sessions". This approach provides for timely and constructive criticism, but it has also caused an increase in the effort expended in communication, resolving problems, and making the software development visible. Since the acquirer was not willing to relinguish full control and visibility, additional planning and oversight activities have been required.

Reviews

Periodic reviews were agreed upon by both the acquirer and developer, typically at six week intervals. Although the material requires less than traditional formality and preparation, there is still an issue with expectations from both sides. The acquirer, familiar with traditional reviews, still expects completed products. Operationally, these reviews still look like formal reviews. Materials are required in advance, minutes are taken and action items are tracked. So in essence, these informal reviews have consumed nearly the same preparation and presentation time as traditional reviews, and because they are more frequent, ultimately more effort overall.

Documentation

Since plans are the only deliverables "formally approved" by the acquirer, they have been subjected to a large amount of scrutiny. For supportability reasons, the acquirer is interested in other development documentation generated as a record of design decision making. Additional configuration management provisions have been taken to assure that all software products, not just source code, are managed and controlled. Even the developer's notes, that support design decisions, are carefully controlled. Therefore, the effort associated with delivered documentation seems to have increased.

Development Approach

The developer has the flexibility to tailor development activities. In this case, non-waterfall activities such as reengineering and rapid prototyping were planned before or in parallel with requirements engineering. Also, the SW quality assurance team works with the development team during the development of software products, rather than acting as a gate at the end. The developer still needs to develop plans in advance, and they still must be documented.

Summary

To successfully execute a development effort using 498, the organization should have standard processes, tools and methods already in place and prior tailoring experience. If an organization does not have a clearly defined and mature development process as well as prior tailoring experience, the risks of increasing effort, or worse, delivering a product that does not meet requirements, is great when utilizing 498.

RECOMMENDATION

Based on the general consensus that 498 requires a mature development process to be successful, NCCA recommends that no reduction be taken for MIL-STD-498 unless the contractor's performance at an SEI level 3 or higher has been demonstrated (by some methodology other than self-assessment) and he has also demonstrated a savings solely due to the utilization of 498 on at least one previous software development effort.

As historical data becomes available, the NCCA SW team will periodically revisit and update this recommendation.

REFERENCES

1. Radatz, Jane, Myrna Olson and Stuart Campbell, "MIL-STD-498," *CROSSTALK*, STSC, Hill Air Force Base, Utah, February 1995, pp. 2-5, 28.

2. Sorensen, Reed, "Adopting MIL-STD-498: The Steppingstone to the U.S. Commercial Standard," *CROSSTALK*, STSC, Hill Air Force Base, Utah, March 1996, pp. 8-12.

3. Sorensen, Reed, "A Comparison of Software Development Methodologies," *CROSSTALK*, STSC, Hill Air Force Base, Utah, January 1995, pp. 12-18.

4. Baker, Lawrence I., "Details/Implications of MIL-STD-498 (Software Development & Documentation)," paper presented April 1995 at DSMC.

5. Newberry, Maj. George A., "Changes from DOD-STD-2167A to MIL-STD-498," *CROSSTALK*, STSC, Hill Air Force Base, Utah, April 1995, pp. 4-7.

6. Sorensen, Reed, "How is MIL-STD-498 Being Used?," *CROSSTALK*, STSC, Hill Air Force Base, Utah, October 1995, pp. 23-24.

7. Guidelines for Successful Acquisition and Management of Software Intensive Systems: Weapons Systems, Command and Control Systems, Management Information Systems, Version 1.1, STSC, February 1995, pp. 7-13 & 7-14.

8. Szulewski, Paul A. and Maibor, David S., "MIL-STD-498: What's New and Some Real Lessons Learned," *CROSSTALK*, STSC, Hill Air Force Base, Utah, March 1996, pp. 13-16, 23.

9. MIL-STD-498, 498 Application Workshop, Release 1.8, Naval Information Systems Management Center, July 1994.